

Re: understanding: synchronized reference

Source: <http://coding.derkeiler.com/Archive/Java/comp.lang.java.programmer/2006-08/msg00542.html>

- *From:* Robert Klemme <shortcutter@xxxxxxxxxxxxxxxx>
 - *Date:* Wed, 09 Aug 2006 13:23:45 +0200
-

On 09.08.2006 13:00, John_Woo wrote:

Hi,

I want some classification on understanding the context of synchronized reference:

say

```
LinkedList queue = new LinkedList();
```

```
public Object pop() throws InterruptedException {  
    synchronized(queue) {  
        while (queue.isEmpty()) {  
            queue.wait();  
        }  
        return queue.removeFirst();  
    }  
}
```

```
public void put(Object o){queue.add(o);}
```

```
public void do(){  
    pup();  
    put(o);  
    put(o);  
    pup();  
    ..  
}
```

case 1: pop never called

case 2: pop called only once

case 3: pop/put called many times

I'm wondering, in above 3 cases, what happened to queue in terms of synchronization?

like,

in case 1, is it queue not synchronized?

in case 2, if queue size always > 0 ; queue not synchronized?

Re: understanding: synchronized reference

The question doesn't really make sense to me. It reads as if "synchronizing" does something to the queue content which it does not. Synchronizing ensures exclusive access to a lock for a single thread.

If queue should be accessed from multiple threads access needs to be synchronized. Note, that in Java 1.5 there is are queue classes that has proper synchronization built in. See here for example

<http://java.sun.com/j2se/1.5.0/docs/api/java/util/concurrent/BlockingQueue.html>

when queue is synchronized in pop, did that imply it synchronized anywhere in same class?

No. But it must be synchronized in put. Also, put must invoke queue.notifyAll() to make pop() wake up.

Kind regards

robert

.